

# Magnifying Lens Abstraction for Stochastic Games with Discounted and Long-run Average Objectives

Krishnendu Chatterjee<sup>1</sup>, Luca de Alfaro<sup>2</sup>, and Pritam Roy<sup>3</sup>

<sup>1</sup> IST Austria

<sup>2</sup> UC Santa Cruz, USA

<sup>3</sup> UC Los Angeles, USA

**Abstract.** Turn-based stochastic games and its important subclass Markov decision processes (MDPs) provide models for systems with both probabilistic and nondeterministic behaviors. We consider turn-based stochastic games with two classical quantitative objectives: discounted-sum and long-run average objectives. The game models and the quantitative objectives are widely used in probabilistic verification, planning, optimal inventory control, network protocol and performance analysis. Games and MDPs that model realistic systems often have very large state spaces, and probabilistic abstraction techniques are necessary to handle the state-space explosion. The commonly used full-abstraction techniques do not yield space-savings for systems that have many states with similar value, but does not necessarily have similar transition structure. A semi-abstraction technique, namely Magnifying-lens abstractions (MLA), that clusters states based on value only, disregarding differences in their transition relation was proposed for qualitative objectives (reachability and safety objectives) [8]. In this paper we extend the MLA technique to solve stochastic games with discounted-sum and long-run average objectives. We present the MLA technique based abstraction-refinement algorithm for stochastic games and MDPs with discounted-sum objectives. For long-run average objectives, our solution works for all MDPs and a sub-class of stochastic games where every state has the same value.

## 1 Introduction

A *turn-based stochastic game* is played on a finite graph with three types of states: in player-1 states, the first player chooses a successor state from a given set of outgoing edges; in player-2 states, the second player chooses a successor state from a given set of outgoing edges; and in probabilistic states, the successor state is chosen according to a given probability distribution. The game results in an infinite path through the graph. An important subclass of turn-based stochastic games is *Markov decision processes (MDPs)*: in MDPs the set of player-2 states is empty. Turn-based stochastic games and MDPs provide models for the study of dynamic systems that exhibit both probabilistic and nondeterministic behavior.

Turn-based stochastic games with *qualitative objectives* such as reachability, safety, and more general  $\omega$ -regular objectives has been widely studied in literature [6,7,2,5] in the context of verification of probabilistic systems. Many other application scenarios such as planning, inventory control, performance analysis require the study of turn-based stochastic games with *quantitative objectives* [9,1,17,7,16]. The two classical quantitative objectives studied in literature are as follows: *discounted-sum* (in short, discounted) and *long-run average* objectives [10,1]. In both these objectives a real-valued reward is assigned to every state. For an infinite path (infinite sequence of states in the game graph), the discounted objective assigns a payoff that is the discounted sum of the rewards that appear in the infinite path, and the long-run average objective assigns the long-run average of the rewards that appear in the path. Turn-based stochastic games and MDPs with discounted and long-run average objectives provide an important and powerful framework for studying a wide range of applications [10,1].

Turn-based stochastic games and MDPs that model realistic systems typically have very large state spaces. Therefore the main algorithmic challenge in analysing such models consist of developing algorithms that work efficiently on large state spaces. In the non-probabilistic setting, abstraction techniques

have been successful in coping with large state-spaces [4]. By ignoring details not relevant to the property under study, abstraction makes it possible to answer questions about a system through the analysis of a smaller, more concise abstract model. The abstraction-refinement techniques for non-probabilistic setting do not always have a straight-forward extension to the probabilistic models. The commonly used full-abstraction techniques do not yield space-savings for systems that have many states with similar value, but not necessarily have similar transition structure. A semi-abstraction technique, namely *Magnifying-lens abstractions* (MLA), was proposed for a subclass of qualitative objectives (namely, reachability and safety objectives) [8]. MLA is a semi-abstract technique that can cluster states based on value only and can disregard the differences in their transition relation. MLA is particularly well-suited to problems where there is a notion of *locality* in the state space, so that it is useful to cluster states based on values, even though their transition relations may not be similar. Many inventory, planning and control problems satisfy the locality property and would benefit from the MLA technique. In the setting of inventory, planning and control problems quantitative objectives are more appropriate than qualitative objectives. This provides a strong and practical motivation for extending the work of [8] to provide MLA technique based solution for turn-based stochastic games and MDPs with quantitative objectives.

In this paper we extend the MLA technique to solve stochastic games with quantitative objectives. The MLA technique of [8] works for MDPs and the special class of qualitative objectives, namely reachability and safety objectives (the model is quantitative with probabilities but the objectives are qualitative). We present the MLA technique based abstraction-refinement algorithm for both stochastic games and MDPs with discounted objectives. For long-run average objectives, our solution works for all MDPs and a subclass of stochastic games where every state has the same value. We note that for long-run average objectives in stochastic games, the same assumption (of all states having the same value) is required for the *relative value* iteration algorithm to work [1,10]<sup>4</sup>. Hence our result present generalizations of the results of [8] from the sub-class of reachability and safety objectives (which are Boolean) to the general class of discounted and long-run average objectives (which are quantitative). An abstraction-refinement based technique was proposed in [3] for turn-based stochastic games with quantitative objectives, but the technique of [3] does not provide either a useful way to abstract probabilities, or the space-saving benefit of the MLA based technique. Thus our algorithms provide space-efficient and practical algorithmic solutions for a wide class of problems of interest. To demonstrate the applicability of our algorithms we present a symbolic implementation of our algorithms for MDPs with discounted objectives. In Section 5 we present many examples to illustrate cases where MLA based solution has a clear advantage over the full abstraction techniques, and our experimental results show that the MLA based technique gives a significant space saving.

## 2 Preliminaries

For a finite set  $S$ , a *probability distribution* on  $S$  is a function  $p : S \rightarrow [0, 1]$  such that  $\sum_{s \in S} p(s) = 1$ ; we denote the set of probability distributions on  $S$  by  $\text{Dist}(S)$ . A *valuation* over a set  $S$  is a function  $v : S \rightarrow \mathbb{R}$  associating a real number  $v(s)$  with every  $s \in S$ . For  $x \in \mathbb{R}$ , we denote by  $\mathbf{x}$  the valuation with constant value  $x$ ; for  $T \subseteq S$ , we indicate by  $[T]$  the valuation having value 1 in  $T$  and 0 elsewhere. For two valuations  $v, u$  on  $S$ , we define  $\|v - u\| = \sup_{s \in S} |v(s) - u(s)|$ .

A *partition* of a set  $S$  is a set  $R \subseteq 2^S$ , such that  $\bigcup_{x \in R} \{s | s \in x\} = S$  and  $x \cap x' = \emptyset$  for all  $x \neq x' \in R$ . For  $s \in S$  and a partition  $R$  of  $S$ , we denote by  $[s]_R$  the element  $x \in R$  with  $s \in x$ . We say that a partition  $R$  is *finer* than a partition  $R'$  if for any  $x \in R$  there exists  $x' \in R'$  such that  $x \subseteq x'$ .

We consider the class of turn-based probabilistic games and its important subclass of Markov decision processes (MDPs).

**Game graphs.** A *turn-based probabilistic game graph* ( $2^{1/2}$ -player game graph)  $G = ((S, E), (S_1, S_2, S_P), \delta)$  consists of a directed graph  $(S, E)$ , a partition  $(S_1, S_2, S_P)$  of the finite set

<sup>4</sup> Thus the assumption is necessary even for classical value iteration algorithms and even without abstraction, and hence cannot be avoided in our setting with abstraction.

$S$  of states, and a probabilistic transition function  $\delta: S_P \rightarrow \text{Dist}(S)$ , where  $\text{Dist}(S)$  denotes the set of probability distributions over the state space  $S$ . The states in  $S_1$  are the *player-1* states, where player 1 decides the successor state; the states in  $S_2$  are the *player-2* states, where player 2 decides the successor state; and the states in  $S_P$  are the *probabilistic* states, where the successor state is chosen according to the probabilistic transition function  $\delta$ . We assume that for  $s \in S_P$  and  $t \in S$ , we have  $(s, t) \in E$  iff  $\delta(s)(t) > 0$ , and we often write  $\delta(s, t)$  for  $\delta(s)(t)$ . For technical convenience we assume that every state in the graph  $(S, E)$  has at least one outgoing edge. For a state  $s \in S$ , we write  $E(s)$  to denote the set  $\{t \in S \mid (s, t) \in E\}$  of possible successors. For  $s \in S$  and a partition  $R$  of  $S$ , a region  $r_2 \in R$  is called successor to a region  $r_1 \in R$  if at least one concrete state in  $r_1$  has non-zero probability to reach concrete state(s) in  $r_2$ . The *Markov decision processes* ( $1^{1/2}$ -player game graphs) are the special case of the  $2^{1/2}$ -player game graphs with  $S_1 = \emptyset$  or  $S_2 = \emptyset$ . We refer to the MDPs with  $S_2 = \emptyset$  as *player-1* MDPs, and to the MDPs with  $S_1 = \emptyset$  as *player-2* MDPs.

**Plays and strategies.** An infinite path, or a *play*, of the game graph  $G$  is an infinite sequence  $\omega = \langle s_0, s_1, s_2, \dots \rangle$  of states such that  $(s_k, s_{k+1}) \in E$  for all  $k \in \mathbb{N}$ . We write  $\Omega$  for the set of all plays, and for a state  $s \in S$ , we write  $\Omega_s \subseteq \Omega$  for the set of plays that start from the state  $s$ . A *strategy* for player 1 is a function  $\sigma: S^* \cdot S_1 \rightarrow \text{Dist}(S)$  that assigns a probability distribution to all finite sequences  $w \in S^* \cdot S_1$  of states ending in a player-1 state (the sequence represents a prefix of a play). Player 1 follows the strategy  $\sigma$  if in each player-1 move, given that the current history of the game is  $w \in S^* \cdot S_1$ , she chooses the next state according to the probability distribution  $\sigma(w)$ . A strategy must prescribe only available moves, i.e., for all  $w \in S^*$ ,  $s \in S_1$ , and  $t \in S$ , if  $\sigma(w \cdot s)(t) > 0$ , then  $(s, t) \in E$ . The strategies for player 2 are defined analogously. We denote by  $\Sigma$  and  $\Pi$  the set of all strategies for player 1 and player 2, respectively.

Once a starting state  $s \in S$  and strategies  $\sigma \in \Sigma$  and  $\pi \in \Pi$  for the two players are fixed, the outcome of the game is a random walk  $\omega_s^{\sigma, \pi}$  for which the probabilities of events are uniquely defined, where an *event*  $\mathcal{A} \subseteq \Omega$  is a measurable set of plays. For a state  $s \in S$  and an event  $\mathcal{A} \subseteq \Omega$ , we write  $\Pr_s^{\sigma, \pi}(\mathcal{A})$  for the probability that a play belongs to  $\mathcal{A}$  if the game starts from the state  $s$  and the players follow the strategies  $\sigma$  and  $\pi$ , respectively. For a measurable function  $f: \Omega \rightarrow \mathbb{R}$  we denote by  $\mathbb{E}_s^{\sigma, \pi}[f]$  the *expectation* of the function  $f$  under the probability measure  $\Pr_s^{\sigma, \pi}(\cdot)$ .

Strategies that do not use randomization are called pure. A player-1 strategy  $\sigma$  is *pure* if for all  $w \in S^*$  and  $s \in S_1$ , there is a state  $t \in S$  such that  $\sigma(w \cdot s)(t) = 1$ . A *memoryless* player-1 strategy does not depend on the history of the play but only on the current state; i.e., for all  $w, w' \in S^*$  and for all  $s \in S_1$  we have  $\sigma(w \cdot s) = \sigma(w' \cdot s)$ . A memoryless strategy can be represented as a function  $\sigma: S_1 \rightarrow \text{Dist}(S)$ . A *pure memoryless strategy* is a strategy that is both pure and memoryless. A pure memoryless strategy for player 1 can be represented as a function  $\sigma: S_1 \rightarrow S$ . We denote by  $\Sigma^{PM}$  the set of pure memoryless strategies for player 1. The pure memoryless player-2 strategies  $\Pi^{PM}$  are defined analogously.

**Quantitative objectives.** A *quantitative* objective is specified as a measurable function  $f: \Omega \rightarrow \mathbb{R}$ . We consider *zero-sum* games, i.e., games that are strictly competitive. In zero-sum games the objectives of the players are functions  $f$  and  $-f$ , respectively. We consider two classical quantitative objectives specified as discounted sum objective and long-run average (mean-payoff) objectives. The definitions of are as follows.

- *Discounted objectives.* Let  $r: S \rightarrow \mathbb{R}_{\geq 0}$  be a real-valued reward function that assigns to every state  $s$  the reward  $r(s)$ , and let  $0 < \beta < 1$  be a discount factor. The *discounted* objective  $\text{Disc}$  assigns to every play the  $\beta$ -discounted sum of the rewards that appears in the play. Formally, for a play  $\omega = \langle s_0, s_1, s_2, s_3, \dots \rangle$  we have  $\text{Disc}(\beta, r)(\omega) = \sum_{i=0}^{\infty} \beta^i \cdot r(s_i)$ .
- *Long-run average objectives.* Let  $r: S \rightarrow \mathbb{R}_{\geq 0}$  be a real-valued reward function that assigns to every state  $s$  the reward  $r(s)$ . The *long-run average* objective  $\text{LimAvg}$  assigns to every play the long-run average of the rewards that appear in the play. Formally, for a play  $\omega = \langle s_1, s_2, s_3, \dots \rangle$  we have  $\text{LimAvg}(r)(\omega) = \liminf_{T \rightarrow \infty} \frac{1}{T} \cdot \sum_{i=0}^{T-1} r(s_i)$ .

**Values and optimal strategies.** Given a game graph  $G$ , and quantitative objectives specified as measurable functions  $f$  and  $-f$  for player 1 and player 2, respectively, we define the *value* functions  $\text{Val}_1$  and  $\text{Val}_2$  for

the players 1 and 2, respectively, as the following functions from the state space  $S$  to the set  $\mathbb{R}$  of reals: for all states  $s \in S$ , let

$$Val_1^G(f)(s) = \sup_{\sigma \in \Sigma} \inf_{\pi \in \Pi} \mathbb{E}_s^{\sigma, \pi}[f]; \quad Val_2^G(-f)(s) = \sup_{\pi \in \Pi} \inf_{\sigma \in \Sigma} \mathbb{E}_s^{\sigma, \pi}[-f].$$

In other words, the values  $Val_1^G(f)(s)$  give the maximal expectation with which player 1 can achieve her objective  $f$  from state  $s$ , and analogously for player 2. The strategies that achieve the values are called optimal: a strategy  $\sigma$  for player 1 is *optimal* from the state  $s$  for the objective  $f$  if  $Val_1^G(f)(s) = \inf_{\pi \in \Pi} \mathbb{E}_s^{\sigma, \pi}[f]$ . The optimal strategies for player 2 are defined analogously. We now state the classical memoryless determinacy results for  $2^{1/2}$ -player games with discounted and long-run average objectives.

**Theorem 1 (Quantitative determinacy [10,12]).** *For all  $2^{1/2}$ -player game graphs  $G$ , the following assertions hold.*

- For all reward functions  $r : S \rightarrow \mathbb{R}_{\geq 0}$ , for all  $0 < \beta < 1$ , and all states  $s \in S$ , we have

$$Val_1^G(\text{Disc}(\beta, r))(s) + Val_2^G(\text{Disc}(\beta, -r))(s) = 0;$$

$$Val_1^G(\text{LimAvg}(r))(s) + Val_2^G(\text{LimAvg}(-r))(s) = 0.$$

- Pure memoryless optimal strategies exist for both players from all states for discounted and long-run average objectives.

We now present the definition of the *predecessor* operator  $\text{Pre}$ . The operator  $\text{Pre}$  is an important operator that is used in many classical algorithms to solve  $2^{1/2}$ -player games with discounted and long-run average objectives.

**Definition 1 (The predecessor operator ( $\text{Pre}$ )).** *Given a game graph  $G = ((S, E), (S_1, S_2, S_P), \delta)$ , the predecessor operator  $\text{Pre}$  takes a valuation  $v : S \rightarrow \mathbb{R}_{\geq 0}$  and returns a valuation  $\text{Pre}(v) : S \rightarrow \mathbb{R}_{\geq 0}$  defined as follows: for every state  $s \in S$  we have*

$$\text{Pre}(v)(s) = \begin{cases} \max_{t \in E(s)} v(t) & s \in S_1 \\ \min_{t \in E(s)} v(t) & s \in S_2 \\ \sum_{t \in S} \delta(s, t) \cdot v(t) & s \in S_P. \end{cases}$$

### 3 MLA for Discounted Objectives

In this section we present algorithmic solutions for  $2^{1/2}$ -player games and MDPs with discounted objectives.

**Classical Algorithms.** We present the algorithms to solve a turn-based stochastic games with discounted objectives.

**Theorem 2 ([10,1]).** *Given a turn-based stochastic game graph  $G$ , with a reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$  and a discount factor  $0 < \beta < 1$ , the following assertions hold.*

1. (Value iteration). *Consider the sequence of valuations  $v_0, v_1, v_2, \dots$  as follows: let  $v_0 = \mathbf{0}$  and for all  $i \geq 0$  and  $s \in S$  we have*

$$v_{i+1}(s) = (1 - \beta) \cdot r(s) + \beta \cdot \text{Pre}(v_i)(s).$$

*The sequence  $(v_i)_{i \geq 0}$  converges monotonically to  $Val_1^G(\text{Disc}(\beta, r))$ .*

2. (Fixpoint solution). *There exists a valuation  $v^*$  that is the unique fixpoint of the function  $f(v)(s) = (1 - \beta) \cdot r(s) + \beta \cdot \text{Pre}(v)(s)$ , i.e., for all  $s \in S$  we have*

$$v^*(s) = (1 - \beta) \cdot r(s) + \beta \cdot \text{Pre}(v^*)(s)$$

and we have  $v^* = \text{Val}_1^G(\text{Disc}(\beta, r))$ .

*The classical algorithms.* The classical algorithms for solving turn-based stochastic games are based on the result of Theorem 2 and are as follows.

1. We obtain the sequence of valuations  $(v_i)_{i \geq 0}$  as given by Theorem 2 by iterating over the valuations, and the sequence converges (w.r.t. an error tolerance  $\varepsilon_{\text{float}}$ ) to the desired value of the game.
2. The fixpoint  $v^*$  that gives the desired value of the game can be obtained by solving optimization problems: if the game graph is an MDP, then it can be obtained from the solution of a linear-programming problem [13], and for general turn-based stochastic games it can be obtained as a solution of a quadratic programming problem [14].

*Abstract properties for upper and lower bound of value functions.* We first present certain abstract properties of functions that can be used to obtain upper and lower bounds on the value of stochastic game with a discounted objectives. Later we will present a concrete functions that satisfies the abstract properties and can be implemented by the magnifying lens abstraction techniques.

**Theorem 3.** *Let  $G$  be turn-based stochastic game graph with reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$  and discount factor  $\beta$ . Let  $M = \max_{s \in S} |r(s)|$  and let  $Q = \frac{M}{1-\beta}$ . Consider the function  $f$  on valuations such that*

$$f(v)(s) = (1 - \beta) \cdot r(s) + \beta \cdot \text{Pre}(v)(s).$$

*Let  $f_+$  and  $f_-$  be two functions on valuations that satisfy the following conditions:*

1.  $f_+$  and  $f_-$  are monotonic;
2. for all valuations  $v$  we have  $f_-(v) \leq f(v) \leq f_+(v)$ ;
3. for all valuations bounded by  $Q$  (i.e., for all  $s \in S$  we have  $-Q \leq v(s) \leq Q$ ) we have  $-Q \leq f_-(v) \leq f_+(v) \leq Q$ .

*Then there exist least fixpoints  $v_+^*$  and  $v_-^*$  of  $f_+$  and  $f_-$  and  $v_-^* \leq \text{Val}_1^G(\text{Disc}(\beta, r)) \leq v_+^*$ .*

In the following we will use the magnifying lens abstraction techniques to define functions  $f_+$  and  $f_-$  that satisfies the properties of the above theorem. This will allow us to obtain efficient solution of turn-based stochastic games with abstraction techniques.

**Magnifying Lens Abstraction Algorithm.** *Magnifying-lens abstractions* (MLA) is a semi-abstract technique that can cluster states based on value only, disregarding differences in their transition relation. Let  $v^*$  be the discounted sum valuation over  $S$  that is to be computed. Given a desired accuracy  $\varepsilon_{\text{abs}} > 0$ , MLA computes upper and lower bounds for  $v^*$ , spaced less than  $\varepsilon_{\text{abs}}$  apart.

*Algorithm Sketch.* The MLA algorithm is shown in Algorithm 1. The algorithm has parameters  $G, \beta, r$ , and errors  $\varepsilon_{\text{abs}} > 0, \varepsilon_{\text{float}} \geq 0$ . Parameter  $\varepsilon_{\text{abs}}$  indicates the allowed maximum difference between the lower and upper bounds returned by MLA. MLA starts from an initial partition (set of regions)  $R$  of  $S$ . The initial partition  $R$  is obtained either from the user or from the property. Statement 2 initializes the valuations  $u^-$  and  $u^+$  to  $\mathbf{0}$  since discounted sums are computed as least fixpoints. MLA computes the lower and upper bounds as valuations  $u^-$  and  $u^+$  over  $R$  by GlobalValIter Algorithm (Algorithm 2). Global iterations, when implemented as a value iteration (Algorithm 2), contains an extra parameter  $\varepsilon_{\text{float}} > 0$ . Parameter  $\varepsilon_{\text{float}}$ , stopping parameter of classical value iteration, specifies the degree of precision to which the global value iteration should converge. For accurate global iterations, we can set the parameter  $\varepsilon_{\text{float}}$  to 0. The partition is refined, until the difference between  $u^-$  and  $u^+$ , for all regions, is below a specified threshold.

---

**Algorithm 1**  $\text{MLA}(G, \beta, r, \varepsilon_{abs}, \varepsilon_{float})$  Magnifying-Lens Abstraction

---

**Input** : game  $G$ , discount factor  $\beta$ ,  
reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$ ,  
errors  $\varepsilon_{abs} > 0, \varepsilon_{float} \geq 0$   
**Output** : final partition  $R$ , valuations  $u^+, u^- : R \rightarrow \mathbb{R}_{\geq 0}$

1.  $R :=$  some initial partition.
2.  $u^- := 0; u^+ := 0$
3. **loop**
4.    $u^+ := u^-$
5.    $u^+ := \text{GlobalValIter}(G, R, u^+, \beta, r, \text{max}, \varepsilon_{float})$
6.    $u^- := \text{GlobalValIter}(G, R, u^-, \beta, r, \text{min}, \varepsilon_{float})$
7.   **if**  $\|u^+ - u^-\| \geq \varepsilon_{abs}$
8.     **then**  $R, u^-, u^+ := \text{SplitRegions}(R, u^-, u^+, \varepsilon_{abs})$
9.     **else return**  $R, u^-, u^+$
10.   **end if**
11. **end loop**

---

---

**Algorithm 2**  $\text{GlobalValIter}(G, R, u, \beta, r, h, \varepsilon_{float})$  Global Value Iteration

---

**Input** : game  $G$ , partition  $R$ , valuation  $u : R \rightarrow \mathbb{R}_{\geq 0}$ ,  
discount factor  $\beta$ , reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$ ,  
 $h \in \{\text{max}, \text{min}\}$ , error  $\varepsilon_{float} \geq 0$   
**Output** : valuation  $u : R \rightarrow \mathbb{R}_{\geq 0}$

1. **repeat**
2.    $\hat{u} := u$
3.   **for**  $x \in R$  **do**
4.      $u(x) := \text{MagIter}(G, R, x, \hat{u}, \beta, r, h, \varepsilon_{float})$
5.   **end for**
6. **until**  $\|u - \hat{u}\| \leq \varepsilon_{float}$
7. **return**  $u$

---

*Global Value Iteration (GlobalValIter).* To compute  $u^-$  (resp.  $u^+$ ), GlobalValIter considers each region  $x \in R$  in turn, and performs a *magnified iteration (MI)*: it improves the bounds  $u^-(x)$  (resp.  $u^+(x)$ ) by solving the sub-games on the concrete states in  $r$ .

*MagnifiedIteration (MagIter).* The goal of the magnified iteration algorithm is to either (a) iterate function  $f_+$  and  $f_-$  with properties of Theorem 3 or (b) obtain fixpoints of  $f_+$  and  $f_-$ . To obtain the desired functions we define an auxiliary function  $g$  and a magnified predecessor operator  $MPre$  and then present the magnifying lens abstraction implementation of  $MPre$ .

**Definition 2.** Given a game graph  $G = ((S, E), (S_1, S_2, S_P), \delta)$ , two states  $s, t \in S$ , a partition  $R$ , a valuation  $v : S \rightarrow \mathbb{R}_{\geq 0}$ ,  $h \in \{\text{max}, \text{min}\}$ , we define the following auxiliary function  $g$  as follows:

$$g(s, h, R, v)(t) = \begin{cases} v(t) & t \in [s]_R \\ h\{v(t') \mid t' \in [t]_R\} & t \notin [s]_R \end{cases}$$

The function  $g$  is as follows: given two states  $s$  and  $t$ , a valuation  $v$ , a partition  $R$  and a function  $h \in \{\text{max}, \text{min}\}$ , it returns the valuation  $v(t)$  if  $s$  and  $t$  belong to the same partition, otherwise it returns the result of applying  $h$  to the values  $v(t')$  of the states  $t'$  that belongs to the same region as  $t$ . We now define the magnified predecessor operator  $MPre$  that is similar to  $Pre$  but applies the function  $g$  to obtain values.

**Definition 3 (Magnified Predecessor Operator (MPre)).** Given a game graph  $G = ((S, E), (S_1, S_2, S_P), \delta)$ , a partition  $R$ , a valuation  $v : S \rightarrow \mathbb{R}_{\geq 0}$ ,  $h \in \{\text{max}, \text{min}\}$ , we define the

valuation  $MPre(h, v, R) : S \rightarrow \mathbb{R}_{\geq 0}$  as follows: let  $z$  represent  $(s, h, R, v)$ , then for all states  $s \in S$ , we have

$$MPre(h, v, R)(s) = \begin{cases} \max_{t \in E(s)} g(z)(t) & s \in S_1 \\ \min_{t \in E(s)} g(z)(t) & s \in S_2 \\ \sum_{t \in E(s)} \delta(s, t) \cdot g(z)(t) & s \in S_p \end{cases}$$

**Lemma 1 (Properties of  $MPre$ ).** *Given a game graph  $G$ , for all partitions  $R$  and all  $h \in \{\max, \min\}$ , we have*

1.  $MPre(h, v, R)$  is monotonic i.e. for two valuations  $v, v'$ , if  $v \leq v'$ , then  $MPre(h, v, R) \leq MPre(h, v', R)$ .
2. If valuation  $v$  is bounded by  $Q$ , then  $MPre(h, v, R)$  is also bounded by  $Q$ .
3. If  $h$  is max, then  $Pre(v) \leq MPre(h, v, R)$ , and if  $h$  is min, then  $Pre(v) \geq MPre(h, v, R)$ .

The above lemma shows that  $MPre$  with  $h$  as max and min, respectively, satisfies all the properties of  $f_+$  and  $f_-$  of Theorem 3, respectively. Hence we obtain the following lemma.

**Lemma 2.** *Given a game  $G$ , for all partitions  $R$ , and all valuations  $v : S \rightarrow \mathbb{R}_{\geq 0}$ , consider the following functions:*

$$\begin{aligned} l_+(v)(s) &= \beta \cdot r(s) + (1 - \beta) \cdot MPre(\max, v, R)(s); \\ l_-(v)(s) &= \beta \cdot r(s) + (1 - \beta) \cdot MPre(\min, v, R)(s). \end{aligned}$$

*Then there exist least fixpoints  $v_+^*$  and  $v_-^*$  of  $l_+$  and  $l_-$ , respectively, such that  $v_-^* \leq Val_1^G(\text{Disc}(\beta, r)) \leq v_+^*$ .*

**Magnified Iteration Implementation.** We now present the implementation details of the magnified iteration techniques. The operator  $MPre$  takes as input a valuation over the whole state-space  $S$ , and returns a valuation over the whole state space. In the magnifying lens abstraction implementation, our goal is to save space, and operate on valuations that are not on the whole state space. To achieve this goal, for a given region  $x \in R$ , we define a new operator  $\widehat{MPre}_x$  and present its relation with  $MPre$ .

**Definition 4 ( $\widehat{MPre}_x$ ).** *Given a game graph  $G = ((S, E), (S_1, S_2, S_p), \delta)$ , a partition  $R$ , a region  $x \in R$ , valuations  $u : R \rightarrow \mathbb{R}_{\geq 0}$ ,  $v_x : x \rightarrow \mathbb{R}_{\geq 0}$ , we define the valuation  $\widehat{MPre}_x(v_x, R, u) : x \rightarrow \mathbb{R}_{\geq 0}$  as follows: for all states  $s \in x$ , we have*

$$\widehat{MPre}_x(v_x, R, u)(s) = \begin{cases} \max_{t \in E(s)} \hat{g}(y)(t) & s \in S_1 \\ \min_{t \in E(s)} \hat{g}(y)(t) & s \in S_2 \\ \sum_{t \in E(s)} \delta(s, t) \cdot \hat{g}(y)(t) & s \in S_p \end{cases}$$

where  $y$  represents  $(s, R, v_x, u)$ . The auxiliary function  $\hat{g}$  can be defined as follows:

$$\hat{g}(s, R, v_x, u)(t) = \begin{cases} v_x(t) & t \in [s]_R \\ u([t]_R) & t \notin [s]_R \end{cases}$$

Observe that  $\widehat{MPre}_x$  takes a valuation on the states of a region  $x$  (instead of a valuation on the whole state space), and a valuation on the partition of the state space (and hence requires much smaller memory than a valuation on the whole state space). The following lemma establishes the relation of  $MPre$  and  $\widehat{MPre}_x$ . Hence we always achieve the implementation of the  $MPre$  operator as  $\widehat{MPre}_x$ .

**Lemma 3 (Relation of  $MPre$  and  $\widehat{MPre}_x$ ).** Given a game graph  $G$ , for all partitions  $R$  and all  $h \in \{\max, \min\}$ , for all valuations  $v : S \rightarrow \mathbb{R}_{\geq 0}$ , for all  $x \in R$ , let  $v_x : x \rightarrow \mathbb{R}_{\geq 0}$  be a valuation such that  $v_x(s) = v(s)$  for all  $s \in x$ , and let  $u : R \rightarrow \mathbb{R}_{\geq 0}$  be a valuation such that  $u(x) = h\{v(s) \mid s \in x\}$ . Then we have  $\widehat{MPre}_x(v_x, R, u)(s) = MPre(h, v, R)(s)$  for all  $s \in x$ .

Magnified iteration, which involves the  $\widehat{MPre}_x$  implementation of  $MPre$  using magnifying-lens abstraction technique, can be done in two ways like the classical algorithms. We present them below.

*Solution of fixpoint by optimization.* The fixpoints of the functions that provide upper and lower bound on the value using  $MPre$  and  $h$  as max and min can be obtained by solution of optimization problems. We present the fixpoint solution for the case when  $h$  is max and the case when  $h$  is min is similar. Given a partition  $R$ , we have two valuation variables  $u^+ : R \rightarrow \mathbb{R}_{\geq 0}$  and  $v : S \rightarrow \mathbb{R}$  and we denote by  $v_x$  the valuation variable  $v$  restricted to a region  $x \in R$ . We have a set of global constraints that specifies that in every region  $x$  the value  $u^+(x)$  is the maximum value of  $v_x(s)$  for all  $s \in x$ ; i.e., we have the following constraints

$$u^+(x) = h_{s \in x} v_x(s) \text{ for all } x \in R.$$

Along with the above constraints we have local constraints for every region  $x \in R$  and it specifies that  $v_x(s)$  should satisfy the fixpoint constraints for  $\widehat{MPre}_x$ . In other words, for every region  $x \in R$  we have the following set of local constraints:

$$v_x(s) = (1 - \beta) \cdot r(s) + \beta \cdot \widehat{MPre}_x(v_x, R, u^+)(s) \text{ for all } s \in x.$$

Thus instead of solving one huge optimization problem, using the  $\widehat{MPre}_x$  we decompose the optimization problem into many smaller sub-problems with independent sub-parts. Thus the solution is more space efficient and can be achieved faster in practice. Also notice that the solution by optimization to obtain the fixpoint correspond to the solution of magnified iteration (MagIter) with  $\varepsilon_{float} = 0$ .

**Theorem 4 (Correctness of Approximation).** Given a turn-based stochastic game  $G = ((S, E), (S_1, S_2, S_P), \delta)$ , a discount factor  $\beta$ , a reward function  $r$ , and error bounds  $\varepsilon_{abs} > 0$ , and  $\varepsilon_{float} = 0$ , the following assertions hold: let  $(R, u^+, u^-) = MLA(G, \beta, r, \varepsilon_{abs}, 0)$ , then

1. for all  $s \in S$  we have  $u^-([s]_R) \leq Val_1^G(Disc(\beta, r))(s) \leq u^+([s]_R)$ ; and
2. for all  $x \in R$  we have  $u^+(x) - u^-(x) \leq \varepsilon_{abs}$ .

*Value iteration implementation of MagIter.* The Magnified Iteration (MagIter) step can also be implemented as a value iteration approach. When MagIter is implemented as a value iteration, then we require that  $\varepsilon_{float} > 0$ . The parameter  $\varepsilon_{float}$  specifies the degree of precision to which the local, magnified value iteration should converge. Algorithm 3 describes the formal description of the procedure.

**Theorem 5 (Termination and Correctness).** Given a turn-based stochastic game  $G = ((S, E), (S_1, S_2, S_P), \delta)$ , a discount factor  $\beta$ , a reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$ , for all error bounds  $\varepsilon_{abs} > 0$ , the following assertions hold.

1. For all  $\varepsilon_{float} > 0$ , the call  $MLA(G, \beta, r, \varepsilon_{abs}, \varepsilon_{float})$  terminates.
2. There exists an error bound  $\varepsilon_{float}$  such that if  $(R, u^+, u^-) = MLA(G, \beta, r, \varepsilon_{abs}, \varepsilon_{float})$ , then
  - (a) for all  $s \in S$  we have  $u^-([s]_R) \leq Val_1^G(Disc(\beta, r))(s) \leq u^+([s]_R)$ ; and
  - (b) for all  $x \in R$  we have  $u^+(x) - u^-(x) \leq \varepsilon_{abs}$ .

*Adaptive refinement step (SplitRegions).* The step *SplitRegions* is obtained by adaptive refinement of regions with large imprecisions. We denote the *imprecision* of a region  $x$  by  $\Delta(x) = u^+(x) - u^-(x)$ . *MLA* adaptively refines a partition  $R$  by splitting all regions  $x$  having  $\Delta(x) > \varepsilon_{abs}$ . The refinement scheme is simple and easy to implement. Thus a call to *SplitRegions*( $R, u^+, u^-, \varepsilon_{abs}$ ) returns a triple  $\tilde{R}, \tilde{u}^-, \tilde{u}^+$ , consisting of the new partition with its upper and lower bounds for the valuation. Like [8], we also tried other refinement heuristics, but none of them gave strictly better results.



---

**Algorithm 3** MagIter( $G, R, x, u, \beta, r, h, \varepsilon_{float}$ )

---

**Input** : game  $G$ , partition  $R$ , a region  $x \in R$ ,  
valuation  $u : R \rightarrow \mathbb{R}_{\geq 0}$ , discount factor  $\beta$ ,  
reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$   
 $h \in \{max, min\}$ , error  $\varepsilon_{float}$   
**Output** : a value  $u(x) : \mathbb{R}_{\geq 0}$   
**Data Structure** :  $v, \hat{v}$ : valuations over  $x$   
1. **for**  $s \in x$  **do**  $v(s) = u(r)$  **end for**  
2. **repeat**  
3.    $\hat{v} := v$   
3.   **for**  $s \in x$  **do**  
4.      $v(s) = (1 - \beta) \cdot r(s) + \beta \cdot \widehat{MPre}_x(\hat{v}, R, u)(s)$   
5.   **end for**  
6. **until**  $\|v - \hat{v}\| \leq \varepsilon_{float}$   
7. **return**  $h\{v(s) \mid s \in x\}$ 

---

*Space Savings* For value iteration algorithm, the space requirement is equal to the size of state-space  $|S|$ , the domain of  $v$ . For MLA, the space requirement is equal to be the maximum value of  $2 \cdot |R| + \max_{x \in R} |x|$ . The expression gives the maximum space required to store the valuations  $u^+$ ,  $u^-$ , as well as the values  $v$  for the largest magnified region. Since  $\max_{x \in R} |x| \geq (|S|/|R|)$ , the space complexity of the algorithm is (lower) bounded by a square-root function  $\sqrt{8 \cdot |S|}$ . However, this bound is provided for the concrete implementation.

## 4 MLA for Long-run Average Objectives

In this section we present magnifying lens abstraction solution for a class of stochastic games with long-run average objectives. We first describe the efficient classical solution and then present our magnifying lens abstraction solution.

*Value iteration for long-run average objectives.* A value iteration algorithm can be used to compute the long-run average value as follows: for a state  $s$  we compute by value iteration the maximum expected sum of the rewards for  $k$ -step starting from  $s$ , and we denote this sum as  $S(k, s)$ . Then the value of the state  $s$  is  $\lim_{k \rightarrow \infty} \frac{S(k, s)}{k}$ . However, this technique is not very practical as  $S(k, s) \rightarrow \infty$  and  $S(k, s)$  diverges fast towards infinity. Hence computing  $S(k, s)$  and dividing by  $k$  is computationally expensive and not very practical. This problem can be alleviated by *relative value iteration* algorithm that subtracts a number  $c \in \mathbb{R}$  in each iteration. This technique trims the values for all states simultaneously, and this technique is an efficient way to compute values in games that have same values in all states.

**Lemma 4.** Consider a turn-based stochastic game graph  $G = ((S, E), (S_1, S_2, S_P), \delta)$  with a reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$ . For a real number  $c$ , consider a sequence of valuations  $(v_i)_{i \geq 0}$  as follows: let  $v_0 = c$  and for all  $i \geq 0$  and  $s \in S$  we have,  $v_{i+1}(s) = r(s) - c + Pre(v_i)(s)$ . If there exists a real value  $v^*$  such that for all  $s \in S$  we have  $Val_1^G(\text{LimAvg}(r))(s) = v^*$ , then the following conditions hold:

1. The sequence  $(v_i)_{i \geq 0}$  diverges to  $+\infty$  iff  $c < v^*$ .
2. The sequence  $(v_i)_{i \geq 0}$  diverges to  $-\infty$  iff  $c > v^*$ .

The relative value iteration algorithm chooses a real value  $c$ , and then adjusts the value of  $c$  adaptively depending on whether the sequence  $(v_i)_{i \geq 0}$ , given the chosen value  $c$ , diverges to  $+\infty$  or  $-\infty$ , otherwise the chosen real number  $c$  is value of the game.

**MLA for Stochastic Games.** We develop magnifying lens abstraction solution for stochastic games under the assumption that there is a *uniform value*  $v^*$  such that every state has the same value  $v^*$ . Later we will

consider the question of presenting criteria for its existence. For MDPs we will present our solution of all MDPs (without the assumption of existence of uniform value). The magnifying lens abstraction solution for stochastic games with long-run average objective is based on the following lemma.

**Lemma 5 (Magnified Relative Value Iteration.).** *Given a game graph  $G$ , for all partitions  $R$ , consider two sequence of valuations  $(v_i^+)_{i \geq 0}$  and  $(v_i^-)_{i \geq 0}$  as follows : let  $v_0^+ = v_0^- = c$  and for all  $i \geq 0$  and  $s \in S$  we have:*

$$v_{i+1}^+(s) = r(s) - c + MPre(\max, v_i^+, R)(s)$$

$$v_{i+1}^-(s) = r(s) - c + MPre(\min, v_i^-, R)(s)$$

*If there exists a real value  $v^*$  such that for all  $s \in S$  we have  $Val_1^G(\text{LimAvg}(r))(s) = v^*$ , then the following conditions hold:*

1. *If the sequence  $(v_i^-)_{i \geq 0}$  diverges to  $+\infty$ , then  $c < v^*$ .*
2. *If the sequence  $(v_i^+)_{i \geq 0}$  diverges to  $-\infty$ , then  $c > v^*$ .*

Lemma 4 relates the divergence of the sequence  $(v_i)_{i \geq 0}$  for a chosen  $c$  and the value of the game in both directions (iff conditions giving necessary and sufficient conditions), whereas the Lemma 5 (with magnified pre operator) relates the value of  $c$  and the divergence of the sequence in one direction, i.e., if the sequence diverges in a given direction (i.e.,  $(v_i^+)_{i \geq 0}$  to  $-\infty$  or  $(v_i^-)_{i \geq 0}$  to  $+\infty$ ), then we conclude the relation of the value of the game and the chosen value  $c$ . We now present the algorithm that is based on Lemma 5.

*Algorithm Sketch.* Algorithm 4 provides an algorithm to approximate the long-run average value of a stochastic game  $G$  such that every state has the same value. Algorithm uses Lemma 5 to obtain upper and lower bound on the value of the game by a dichotomic (binary) search. The search space is bounded by the interval  $[c^-, c^+]$ , where  $c^+$  and  $c^-$  denote an upper and a lower bound on the value of the game, respectively. The initial value of  $c^+$  (resp.  $c^-$ ) is obtained from the maximum (resp. minimum) reward value of the game. Each iteration of this binary search starts by setting  $c$  to the mid-point of the interval. If with the chosen value of  $c$ , the sequence  $(v_i^+)_{i \geq 0}$  diverges to  $-\infty$ , then  $c$  is an upper bound on the value of the game, and  $c^+$  is set (decreased) to  $c$ . Similarly, if with the chosen value of  $c$ , the sequence  $(v_i^-)_{i \geq 0}$  diverges to  $+\infty$ , then  $c$  is a lower bound on the value of the game and  $c^-$  is set (increased) to  $c$ . The procedure

---

**Algorithm 4** MLALongRun( $G, r, \varepsilon_{abs}, k$ ) Magnifying-Lens Abstraction

---

**Input :** game  $G$ , reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$

errors  $\varepsilon_{abs} > 0$ ,  
maximum number of iterations  $k : \text{integer}$   
a real value  $ratio \in [0, 1]$

**Output :** final partition  $R$

1.  $R :=$  some initial partition
  2.  $c^+ := \max_{s \in S} r(s)$ ,  $c^- := \min_{s \in S} r(s)$
  3. **while**  $(c^+ - c^-) \leq \varepsilon_{abs}$  **do**
  4.    $c := (c^+ + c^-)/2$
  5.    $d^+, v^+ := \text{CheckDivergence}(G, r, R, c, \max, k)$
  6.    $d^-, v^- := \text{CheckDivergence}(G, r, R, c, \min, k)$
  7.   **if**  $d^- = +$  **then**  $c^- := c$
  8.   **else if**  $d^+ = -$  **then**  $c^+ := c$
  8.   **else**  $R := \text{SplitRegions}(R, v^+, v^-, \varepsilon_{abs}, ratio)$
  9.   **end if**
  10. **end while**
- 

*CheckDivergence* returns a verdict on divergence of the sequence by computing  $(k + 1)$  elements of the

sequence. The verdict  $+$ ,  $-$  denote the divergence to  $+\infty$  and  $-\infty$  respectively. However, the verdict  $?$  tells that either (a)  $k$ -elements of the sequence is not enough to detect the divergence or (b) the sequence may not diverge to  $+\infty$  or  $-\infty$ . If *CheckDivergence* returns  $+$  for the  $d^-$ , then  $c^-$  is set to  $c$ , and if *CheckDivergence* returns  $-\infty$  for the  $d^+$ , then  $c^+$  is set to  $c$ . Otherwise we do not have enough information to update  $c^+$  or  $c^-$ , and algorithm refines the partition  $R$  by invoking *SplitRegions* procedure. The *imprecision* of a region  $x \in R$  is denoted by  $\Delta(x) = v^+(x) - v^-(x)$ . The procedure *SplitRegions* splits a number (precisely  $ratio \cdot |R|$ ) of high imprecision regions.

*Detecting divergence to  $+\infty$  and  $-\infty$ .* Algorithm 5 illustrates the procedure *CheckDivergence* to detect the divergence of a sequence starting from a given  $c$  and a given number of iterations  $k$ . If the value for every state increases beyond  $c$  after  $k$ -iterations, then the sequence diverges to  $+\infty$  (the procedure returns  $+$ ), and if the value for every state decreases below  $c$ , then the sequence diverges to  $-\infty$  (the procedure returns  $-$ ). Otherwise, the divergence to  $+\infty$  or  $-\infty$  cannot be concluded and then the procedure returns the verdict  $?$ . The algorithm also returns the  $(k + 1)$ -th valuation of the sequence starting from  $c$ .

---

**Algorithm 5** CheckDivergence( $G, r, R, c, h, k$ )

---

**Input :** game  $G$ , reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$ ,  
a partition  $R$ , a chosen value  $c : \mathbb{R}_{\geq 0}$ ,  
 $h \in \{\max, \min\}$ ,  
maximum number of iterations  $k : integer$   
**Output :** enum  $d \in \{+, -, ?\}$ , valuation  $v : R \rightarrow \mathbb{R}_{\geq 0}$

1.  $v_0 := c, d := ?$
2. **for**  $i = 0$  **to**  $k$  **do**
3.   **for each**  $x \in R$
4.      $v_{i+1}(x) := \text{MagIter2}(G, R, x, v_i, r, h, k)$
5.   **end for**
6. **end for**
7. **if**  $(\min_{x \in R} v_{k+1}(x) > c)$  **then**  $d := +$
8. **if**  $(\max_{x \in R} v_{k+1}(x) < c)$  **then**  $d := -$
9. **return**  $d, v_{k+1}$

---

*Magnified Iteration (Long Run Average version).* Algorithm 6 provides the details of the magnified iteration of a region  $x \in R$ . The algorithm completes value-iteration for  $k$ -iterations over the states of the region  $x$ , and summarizes the values to a single value. Like discounted case, we use  $\widehat{MPre}_x$  operator for the magnifying lens abstraction implementation.

**Theorem 6 (Termination and Correctness).** *Let  $G = ((S, E), (S_1, S_2, S_P), \delta)$  be a turn-based stochastic game with a reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$  such that there exists  $v^* \in \mathbb{R}_{\geq 0}$  and for all  $s \in S$  we have  $Val_1^G(\text{LimAvg}(r))(s) = v^*$ . For all error bounds  $\varepsilon_{abs} > 0$ , the following assertions hold.*

1. *The call  $MLALongRun(G, r, \varepsilon_{abs}, k)$  terminates.*
2. *There exists a positive integer  $k$  such that if  $(R, u^+, u^-) = MLALongRun(G, \beta, r, \varepsilon_{abs}, k)$ , then*
  - (a) *for all  $s \in S$  we have  $u^-(\lfloor s \rfloor_R) \leq Val_1^G(\text{LimAvg}(r))(s) \leq u^+(\lfloor s \rfloor_R)$ ; and*
  - (b) *for all  $x \in R$  we have  $u^+(x) - u^-(x) \leq \varepsilon_{abs}$ .*

*Ensuring uniform value.* The following theorem presents a sufficient condition to ensure uniform value in a turn-based stochastic game (i.e., the same value everywhere). The condition can be checked in polynomial time using algorithms for solving turn-based stochastic reachability games with qualitative winning criteria [5]. For a state  $t$  we denote by  $\Diamond t$  the set of paths that reaches  $t$ .

**Theorem 7.** *Consider a turn-based stochastic game graph  $G$  with a reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$ . Suppose there exists a state  $t$  such that the following conditions hold:*

---

**Algorithm 6** MagIter2( $G, R, x, u, r, h, k$ )

---

**Input** : game  $G$ , partition  $R$ , a region  $x \in R$ ,  
valuation  $u : R \rightarrow \mathbb{R}_{\geq 0}$ ,  
reward function  $r : S \rightarrow \mathbb{R}_{\geq 0}$ ,  $h \in \{\max, \min\}$ ,  
maximum number of iterations  $k : \text{integer}$

**Output** : a value  $u(x) : \mathbb{R}_{\geq 0}$

```
1. for  $s \in x$  do  $v_0(s) = u(r)$  end for
2. for  $i = 0$  to  $k$  do
3.   for  $s \in x$  do
4.      $v_{i+1}(s) := r(s) - c + \widehat{MPre}_x(v_i, R, u)(s)$ 
5.   end for
6. end for
7. return  $h\{v_{k+1}(s) \mid s \in x\}$ 
```

---

1. for all  $s \in S$  there exists a player 1 strategy  $\sigma$  such that against all player 2 strategies  $\pi$  we have  $Pr_s^{\sigma, \pi}(\Diamond t) > 0$ ; and
2. for all  $s \in S$  there exists a player 2 strategy  $\pi$  such that against all player 1 strategies  $\sigma$  we have  $Pr_s^{\sigma, \pi}(\Diamond t) > 0$ .

Then there exists a real value  $v^*$  such that for all  $s \in S$  we have  $Val_1^G(\text{LimAvg}(r)) = v^*$ .

*Proof.* Suppose condition 1 holds, and then by existence of pure memoryless optimal strategies in turn-based stochastic reachability games [5], there is a witness pure memoryless strategy  $\sigma^*$  to witness that for all states  $s$  the state  $t$  is reached with positive probability against all player 2 strategies. Hence if we fix any pure memoryless counter strategy  $\pi$  for player 2 the closed recurrent set must contain  $t$ . From the existence of pure memoryless optimal strategies for turn-based stochastic games with reachability and safety objectives, it follows that player 1 can ensure that from all states  $s$  the state  $t$  is reached with probability 1. Hence for all states  $s$  we have  $Val_1^G(\text{LimAvg}(r))(s) \geq Val_1^G(\text{LimAvg}(r))(t)$ . Similarly, if condition 2 holds, then player 2 can ensure that  $t$  can be reached with probability 1 from all states  $s$  and hence for all states  $s$  we have  $Val_1^G(\text{LimAvg}(r))(s) \leq Val_1^G(\text{LimAvg}(r))(t)$ . Hence  $v^* = Val_1^G(\text{LimAvg}(r))(t)$  is the witness real value to show that the claim holds. ■

**MLA for MDPs.** Now we present the magnifying lens abstraction solution for all MDPs with long-run average objectives (i.e., the solution works for MDPs such that values at different states may be different). The main idea relies on the *end component* decomposition of an MDP.

**Definition 5 (End component).** Given an MDP  $G = ((S, E), (S_1, S_P), \delta)$  a set  $C$  of states is an end component if the following conditions hold: (a) the set  $C$  is strongly connected component in the graph induced by  $(S, E)$ ; and (b) for all probabilistic states  $s \in C \cap S_P$ , all out-going edges of  $s$  is contained in  $C$ , i.e.,  $E(s) \subseteq C$ . An end component  $C$  is maximal if for any end component  $C'$  we have either (a)  $C' \subseteq C$  or (b)  $C' \cap C = \emptyset$ .

The following theorem states that given an MDP with a long-run average objective, if we consider the sub-game graph induced by an end component  $C$ , then all states in  $C$  would have the same value.

**Theorem 8 (Existence of uniform value for MDPs).** Let  $G = ((S, E), (S_1, S_P), \delta)$  be an MDP with a reward function  $r$ . Consider an end component  $C$  in  $G$  and the sub-game graph  $G \upharpoonright C$  induced by  $C$ . Then there exists a real value  $v^*$  such that for all  $s \in C$  we have  $Val_1^{G \upharpoonright C}(\text{LimAvg}(r))(s) = v^*$ .

It follows from Theorem 8 that if we consider the sub-game graph induced by an end component of an MDP, then the condition of uniform value (all states having the same value) is satisfied. It follows from

the results of [7,6] that in an MDP for all strategies with probability 1 the set of states visited infinitely often is an end component. Hence a pure memoryless optimal strategy consists in reaching the correct end component, and then play optimally in the end component. Thus we obtain the following magnifying lens abstraction algorithm for MDPs; the algorithm consists of the following steps:

1. the MDP is decomposed into its maximal end components (this can be achieved in quadratic time);
2. in the sub-game graph induced by an maximal end component we approximate the values by the general algorithm (since the uniform value condition satisfied we can apply the general algorithm for stochastic games);
3. once the values in every maximal end component are approximated we can collapse every maximal end component as a single state and obtain an MDP with no non-trivial end component (every end component is a single state end component), and then compute the values by an algorithm that computes the maximal value that can be reached in an MDP with no non-trivial end components.

## 5 Examples and Experimental Results

In this section, we provide examples and case studies on MDP models with large state-spaces and the locality property. In our implementation we only consider MDPs because common probabilistic model-checkers (like PRISM) only support MDPs. In future work we will consider the stochastic games implementations. Our examples show that the value-based abstraction methods perform better than the transition-based abstraction methods for MDPs with locality property, although for some other case studies the algorithm may provide worse performance. We first present the examples, then our symbolic implementation of MLA algorithms for MDPs with discounted objectives and finally, our experimental results.

Example	Parameters	States	Transitions	non-MLA		MLA		
				Nodes	Time	Nodes	Time	Regions
Planning	n=256,m=40	65,537	265,419	3,981	28	1,658	37	330
	n=512,m=40	262,145	1,063,603	12,420	106	3,324	191	1,121
	n=1024,m=50	1,048,577	4,211,564	15,365	616	4,596	883	1,670
Auto Inventory	$n_{\max}=2,047$ $t_{\max}=2,047$	4,194,304	20,965,376	12,719	65	1,171	209	99
	$n_{\max}=2,047$ $t_{\max}=4,095$	8,388,608	41,930,752	12,676	114	1,095	259	99
	$n_{\max}=4,095$ $t_{\max}=4,095$	16,777,216	83,873,792	25,434	287	6,293	606	99
Machine Replacement	n =1023, tm=1023	1,047,552	3,141,632	6,051	17	419	64	63
	n =2047, tm=2047	4,192,256	12,574,720	12,185	47	960	152	64
	n =4095, tm=4095	16,773,120	50,315,264	24,461	141	960	364	65
Network Protocol	M=7,tm =2,047	1,781,760	7,157,760	328	5	227	11	245
	M = 15, tm=2,047	7,745,536	31,045,632	369	8	267	24	647
	M = 15, tm=4,095	15,491,072	62,091,264	369	17	267	33	647

**Fig. 1.** Experimental results: Symbolic discounted MLA, compared to discounted value iteration

**Planning:** We consider an MDP that models the movement of a robot in a two-dimensional  $(n \times n)$  grid. The grid contains  $m$  mines. The robot at position  $(x, y)$  can choose to move in any of forward, backward, left or right direction to reach the positions  $(x + 1, y)$ ,  $(x - 1, y)$ ,  $(x, y - 1)$  or  $(x, y + 1)$  respectively. However, there is a chance  $p$  that the robot may not reach the desired positions and *dies* due to the explosion of a mine, and in that case the robot reaches a special state called *sink state*. The probability distribution (i.e.,  $p$ ) is a function over distances from the  $m$  mines. The robot spends power for its movement and collects rewards (i.e. recharges) associated with the chargeable states when it visits them. In this example, the state-space is two-dimensional, and every state has transitions to the next states in the state-space (i.e., has the locality property). The robot needs to explore the grid points in an intelligent manner such that the robot spends minimum energy. The property of interest is either to maximize the *discounted reward* or the *long run-average* reward of the robot.

**Automobile Inventory:** In this example we model an inventory of an automobile company. Let  $n$  denote the current number of items in the inventory and  $0 \leq n \leq n_{\max}$  holds where  $n_{\max}$  denotes the maximum capacity of the inventory. Let  $t$  denote the age of the inventory in months and  $0 \leq t \leq t_{\max}$  holds where  $t_{\max}$  is the total life time of the inventory. Let us assume that an unsold car in the inventory becomes cheaper every year by a discount factor as  $\beta$ . Every year company decides whether to manufacture a predefined number  $nc$  of new cars. We assume that the number of cars sold per month, denoted by  $sold$ , follows a uniform probability distribution. We also assume that the value of  $sold$  can vary within a small range  $[sold_{\min}, sold_{\max}]$  and these two constants can be obtained from the car-sale statistics. The reward function is obtained from the cost (negative reward) of manufacturing, and the price (positive reward) of selling a car. The state-space of the model is defined as  $S = \langle n, t \rangle$  where  $n$  denotes the number of the cars and  $t$  denotes the current month. The inventory example contains the locality property; since each state  $(n, t)$  has transitions to the nearby states  $(\langle n - sold, t + 1 \rangle, \langle n, t + 1 \rangle$  or  $\langle n + nc, t + 1 \rangle)$ . After each fiscal year, the company computes the optimal value of the inventory keeping the discount factor in mind. The property of interest is the “optimal discounted sum” of the car inventory.

**Machine Replacement:** In this example we model the machine replacement problem. The state of machine can be in  $n$  different working states from 0 to  $n - 1$ . The state value 0 denotes that the machine is not working and the state value  $n - 1$  denotes the machine is new. The time is denoted by the variable  $t$  and ranges between 0 to a predefined maximum value  $tm$ . The machine can be replaced at any time and the new machine costs money (assume that the machine replacement does not add any reward). The machine can get more work done when it is in a better (state value higher) state, hence earns more money. The property of interest is the “optimal discounted value” of the machine.

**Network Protocol:** Let us assume that  $n$  computers follow a simpler version of Ethernet protocol to send a pre-defined number  $(M - 1)$  of packets to a shared channel. Let  $t$  denotes the time elapsed since the start of the protocol and the condition  $0 \leq t \leq t_{\max}$  holds, where  $t_{\max}$  is the time-out limit of the protocol. The state-space of the model can be given as a tuple  $S = \langle pk_1, pk_2, \dots, pk_n \rangle$  where  $0 \leq pk_i \leq M$  denotes the number of the packets sent from the computer  $i$ . If the computer  $i$  sends one more packet to the channel at time  $t$ , then the  $i$ -th component of the state changes to  $\min\{pk_i + 1, M\}$ . However, two or more computers can send packets to the channel at the same time frame (*collision*) and both packets are lost (the state of MDP does not change). After the collision, each computer waits for a random amount of time before sending it again. Each computer will check whether the channel is busy in time frame  $t$ . If the channel is busy at frame  $t$ , the computer does not send packets at frame  $(t + 1)$ . Otherwise, the computer has two actions - either (1) send at frame  $(t + 1)$ , or (2) does not send. When two computers send packets to the channel at the same time frame, there is a collision and both packets are lost. After the collision, each computer waits for a random amount of time before sending it again. The waiting time for the next packet are decided by the stations following a probability distribution. Since the packets numbers are serial in numbers, the MDP model contains the locality property. The average throughput of the shared channel is measured by the percentage of the packets sent without a collision. We are interested to compute the efficiency of the protocol by computing the average or discounted throughput property.

**MTBDD-based Symbolic Implementation in PRISM :** We have implemented both versions (with and without MLA) of symbolic discounted algorithms within the probabilistic model checker PRISM [11]. We used the MTBDD engine of PRISM, since (a) it is generally the best performing engine for MDPs; and (b) it is the only one that can scale to the size of models we are aiming towards. The current examples with quantitative objectives cannot be handled directly by PRISM, and hence we have added a new functionality of discounted reward computation in the tool PRISM. The initial partitions are picked based on the following choice. Internally, every integer variables with range  $l$  are converted into  $\log_2(l)$  binary variables. If the program have  $k$  binary variables, then we pick  $\frac{k}{2}$  as the initial level of abstraction. We have tried two types of partitioning procedure as proposed in [15].

**Results :** The table above summarizes the results for all case studies (with discount factor 0.9,  $\epsilon_{abs} = 0.01$  and  $\epsilon_{float} = 0.0001$ ). The first two columns show the name and parameters of the MDP model.

The third and fourth columns give the number of states and transitions for each model respectively. The remaining columns show the performance of analyzing the MDPs, using both versions of the discounted algorithms. In both cases, we give the total time required in seconds and the peak MTBDD node count. For MLA, we also show the final number of generated regions. Our results show that MLA algorithm leads to significant space savings which is the real bottleneck in analysis of large MDPs. The number of regions increases with respect to the state-space; however the increase is linear or constant in these examples. The MTBDD node count columns provide a clear view that the symbolic iterations in the value-iteration involve the whole state space and the peak node-count is higher. MLA algorithms computes the value iteration in each region in a sequential manner, hence the size of the MTBDD graph is also smaller. There is a slowdown when MLA is applied; however time is not a bottleneck in the symbolic model-checking tools like PRISM. Most case-studies that PRISM cannot handle often fail due to excessive memory requirements, not due to time. It is also clear, from the sizes of the MDPs in the table, that the symbolic version of MLA is able to handle MDPs considerably larger than were previously feasible for the explicit implementation of [8].

## 6 Conclusion

In this paper we extend the MLA technique to solve MDPs and stochastic games with quantitative objectives. MLA is particularly well-suited to problems where there is a notion of *locality* in the state space, so that it is useful to cluster states based on values, even though their transition relations may not be similar. Many inventory, planning and control problems satisfy the locality property and would benefit from the MLA technique. In the setting of inventory, planning and control problems quantitative objectives are more appropriate than qualitative objectives. We present the MLA technique based abstraction-refinement algorithm for both stochastic games and MDPs with discounted and long run objectives. To demonstrate the applicability of our algorithms we present a symbolic implementation of our algorithms in PRISM for MDPs with discounted objectives. Our experimental results show that the MLA based technique gives significant space saving over value-iteration methods.

## References

1. D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995. Volumes I and II.
2. K. Chatterjee, L. de Alfaro, and T.A. Henzinger. The complexity of stochastic rabin and streett games. In *Proc. 32nd Int. Colloq. Aut. Lang. Prog.*, volume 3580 of *LNCS*, pages 878–890. Springer, 2005.
3. Krishnendu Chatterjee, Thomas A. Henzinger, Ranjit Jhala, and Rupak Majumdar. Counterexample-guided planning. In *UAI*, July 2005.
4. E. Clarke, O. Grumberg, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In *CAV'00*, *LNCS*. Springer, 2000.
5. A. Condon. The complexity of stochastic games. *Information and Computation*, 96:203–224, 1992.
6. C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4):857–907, 1995.
7. L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Stanford University, 1997.
8. Luca de Alfaro and Pritam Roy. Magnifying-lens abstraction for Markov decision processes. In *Proc. CAV'07*, volume 4590 of *LNCS*, pages 325–338. Springer, 2007.
9. C. Derman. *Finite State Markovian Decision Processes*. Academic Press, 1970.
10. J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer, 1997.
11. M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic symbolic model checker. In *TOOLS'02*, volume 2324 of *LNCS*, pages 200–204. Springer, 2002.
12. T. Liggett and S. Lippman. Stochastic games with perfect information and time average payoff. *SIAM Review*, 11:604–607, 1969.
13. A. S. Manne. Linear programming and sequential decisions. *Manag Sci*, 6:259–267, 1960.
14. U.G. Rothblum. Solving stopping stochastic games by maximizing a linear function subject to quadratic constraints. *Game theory and related topics*, 1978.
15. Pritam Roy, David Parker, Gethin Norman, and Luca de Alfaro. Symbolic magnifying lens abstraction in markov decision processes. In *QEST*, pages 103–112, 2008.

16. J. Rutten, M. Kwiatkowska, G. Norman, and D. Parker. *Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems*, volume 23 of *CRM Monograph Series*. AMS, 2004.
17. R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, MIT, 1995.



## Appendix

*Proof. (of Theorem 3).* Since  $f_-$  and  $f_+$  are bounded and monotonic, it follows that there exists fixpoints  $v_-^*$  and  $v_+^*$  of  $f_-$  and  $f_+$ , respectively. Since  $f$  is bounded by  $f_-$  and  $f_+$  it follows that the fixpoint  $v^*$  of  $f$  is bounded by  $v_-^*$  and  $v_+^*$ , respectively, i.e.,  $v_-^* \leq v^* \leq v_+^*$ . The desired result follows. ■

*Proof. (of Lemma 1).* The properties are straightforward to verify using Definition 3. ■

*Proof. (of Lemma 2).* It follows from Lemma 1 that  $l_-$  and  $l_+$  satisfies the properties of functions  $f_-$  and  $f_+$  of Theorem 3, respectively. The results then follows from Theorem 3. ■

*Proof. (of Lemma 3).* The result is easy to using Definition 3 and Definition 4. ■

*Proof. (of Theorem 4 and Theorem 5).* Both the proofs are based on the fact that as  $\varepsilon_{abs}$  and  $\varepsilon_{float}$  converges to 0, the output of the MLA algorithm converges to the value of the game. ■

*Proof. (Lemma 5).* It follows from definition that for all valuations  $v$  we have

$$MPre(\min, v, R) \leq \text{Pre}(v) \leq MPre(\max, v, R).$$

The result follows from the above inequalities and the results of Lemma 4. ■